Dynamic Instruction Scheduling:
Register Renaming

Instructor: Gürhan Küçük

Yeditepe University

A Variation of Tomasulo’s Algorithm
- The dispatch of an instruction with a destination architectural register allocates an available physical register
- Dependent instructions get the data from the allocated physical register (maintains all flow dependencies)
- Each physical register is dynamically mapped to an architectural register (need a mapping table)
- The number of physical registers are greater than the number of architectural registers
- Additional mechanisms are necessary to allocate and deallocate Physical Registers

A Register Renaming Example

Before Register Renaming

After Register Renaming

(RAT_I)

(ARF_I)

(PRF_I)

(PRF_F)

(RAT_F)

(RAT_I)

(RAT_F)

Register Alias Table

Register Alias Table (RAT) or Rename Table (RT) holds the arch.reg to phys.reg. mappings

It is indexed by the architectural register number

Typically, there are separate RATs for integer and FP registers

For example:

RAT_I[5]=40 means integer arch.reg.#5 is mapped to integer phys.reg.#40

Bit vector "Allocated"

Bit vector ‘Allocated’ a.k.a. Allocation List shows the availability of each physical register

If an ‘Allocated’ bit is set, that means, the corresponding phys.reg. is being used by an instruction as a destination register

If an ‘Allocated’ bit is unset, that means, the corresponding phys.reg. is free and can be allocated by incoming instructions
Bit vector “Status”

Bit vector ‘Status’ indicates the status of the physical registers

- If a bit is set in ‘Status’, that means the corresponding phys.reg. has a valid data in its data field
- This field is updated at the time of writeback

Register Renaming (Datapath Components)

- Fetch
- RAT int + FP
- RAT int + FP
- Physical Register Files Int + FP

RR (Cycle 1)

- Fetch LD1
- Decode/RN1
- RAT Int > FP
- PRF Int + FP
- Physical Register Files Int + FP
- RAT_I
- Int FU
- FP Mul/Div
- FP Add
- Load FU
- Store FU
- Common Data Bus

RR (Cycle 2.1)

- Fetch LD2
- Decode/RN1 LD1
- RAT Int > FP
- PRF Int + FP
- Physical Register Files Int + FP
- RAT_I
- Int FU
- FP Mul/Div
- FP Add
- Load FU
- Store FU
- Common Data Bus

Decoding/RN1 for LD F6, R2, #34
RR (Cycle 3)

Meanwhile...
RN2/Dispatch for LD PF0, R2, #34

RR (Cycle 3.2)

Decode/RN1 for MULTD F0, F2, F4

RR (Cycle 4.1)

Decode/RN1 for MULTD F0, F2, F4
RR (Cycle 7.1)

```
LD  Fk, R2, #34
MULT  Fk, F2, F4
DIVD  Fk, F2, F3
ADDD  Fk, F2, F4
```

RR (Cycle 7.2)

```
LD  Fk, R2, #34
MULT  Fk, F2, F4
DIVD  Fk, F2, F3
ADDD  Fk, F2, F4
```

RR (Cycle 8.1)

```
LD  Fk, R2, #34
MULT  Fk, F2, F4
DIVD  Fk, F2, F3
ADDD  Fk, F2, F4
```

RR (Cycle 8.2)

```
LD  Fk, R2, #34
MULT  Fk, F2, F4
DIVD  Fk, F2, F3
ADDD  Fk, F2, F4
```

RR (Cycle 9.1)

```
LD  Fk, R2, #34
MULT  Fk, F2, F4
DIVD  Fk, F2, F3
ADDD  Fk, F2, F4
```

RR (Cycle 9.2)

```
LD  Fk, R2, #34
MULT  Fk, F2, F4
DIVD  Fk, F2, F3
ADDD  Fk, F2, F4
```

© Kucuk'07

PF4 0 PF2 1 PF0
FP Mul/Div
PF5 1 PF0 1P F 1
78*15= ?
PF3 1 PF0 0 PF1
2000+45=2045

ROUTE F

Common Data Bus

A  S        Data A  S     Data
1  0         X  1  0 78
1  1 78
1  0 X
1  1 78
1  0 78
1  0 78
1  0 78
1  0 78
RR (Cycle 58)

LD   $1, R2, #34
LD   $2, R3, #45
MULT $0, R2, #46
SUB  $8, R6, R2
DIV  $10, R0, R6
ADD  $6, R8, R2

Fetch
Decode/RN1
RN2/Dispatch

Int FU      FP Mul/Div      FP Add FU      Load FU      Store FU

Common Data Bus

TAG F

RR (Cycle 59)

LD   $1, R2, #34
LD   $2, R3, #45
MULT $0, R2, #46
SUB  $8, R6, R2
DIV  $10, R0, R6
ADD  $6, R8, R2

Fetch
Decode/RN1
RN2/Dispatch

Int FU      FP Mul/Div      FP Add FU      Load FU      Store FU

Common Data Bus

TAG F